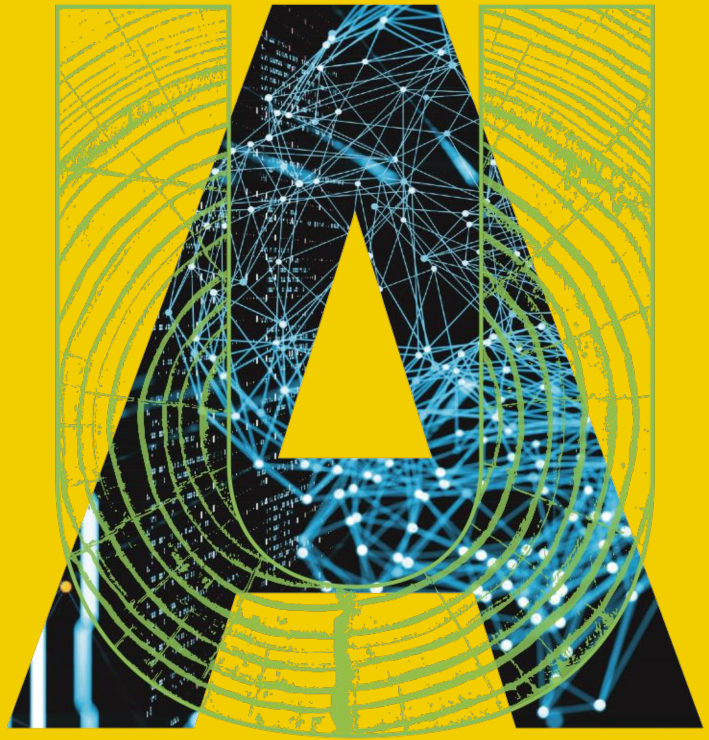
INTRODUCTION TO MACHINE LEARNING

Language Models
Part 1

November 2022
Alex Murphy

UNIVERSITY OF ALBERTA



Intro to ML - Language Models

Topics

- Prehistoric (< 2005) NLP context
- Count-based methods
- Factorisation methods
- Neural Network methods
- Word2Vec (Skip Gram vs CBOW)
- Word vector algebra
- Evaluation metrics (perplexity etc.)
- Beyond “word” vectors
- Recurrent Neural Networks

- Sequence-to-Sequence Models
- Contextualised Embedding Models
- Transfer Learning in NLP
- Transformer Models
- Masked-Language Modelling
- Causal Transformers
- Natural Language Generation
- Sampling / Beam Search
- Multilingual Language Models (time-permitting)
- Language Models beyond Language (time permitting)

What is a Language Model?

A probability distribution over strings according to a model

```
P(string | model)
```

Corpus (plural ***corpora***): a collection of written texts

String:

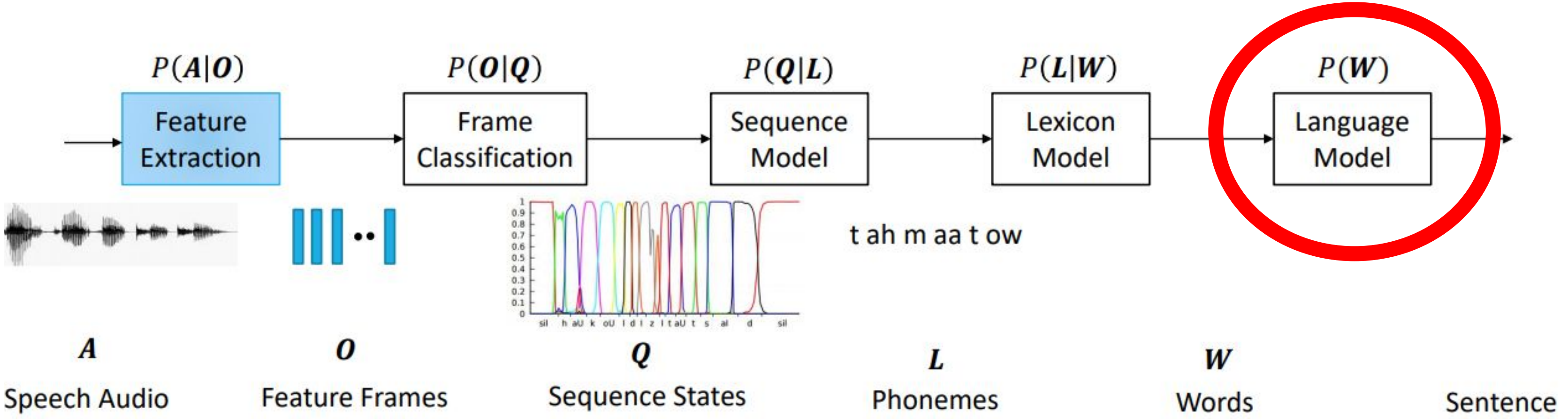
- Word sequence
- Character sequence
- Letter triplets
- Can be most things!

```
P("Edmonton winters are enjoyable" | model)
P("Enjoyable Edmonton are winters" | model)
P("Les hivers d'Edmonton sont plaisants" | model)
```

Language models are defined according to their model and associated model parameters.
Calculation of this value depends on the model, here it's not specified.

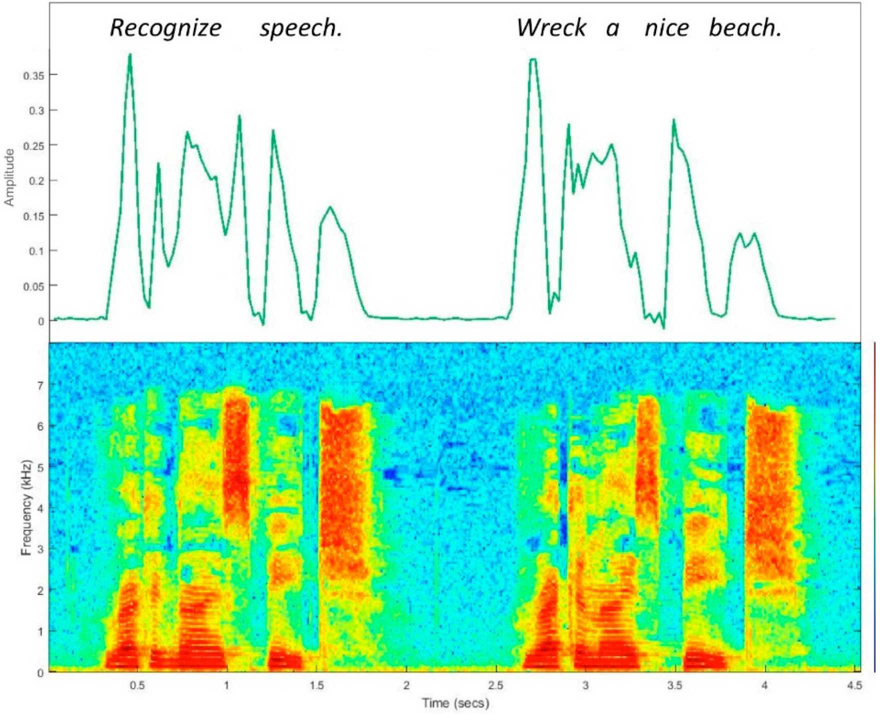
What did we need Language Models for?

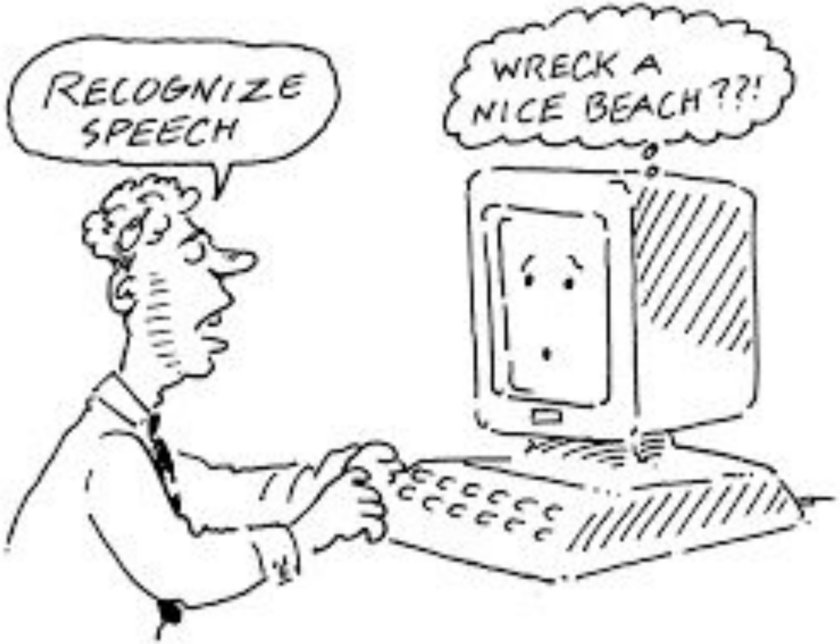
$$\hat{W} = \underset{W}{\operatorname{argmax}} P(W|O) = \underset{W}{\operatorname{argmax}} P(A|O)P(O|Q)P(Q|L)P(L|W)P(W)$$

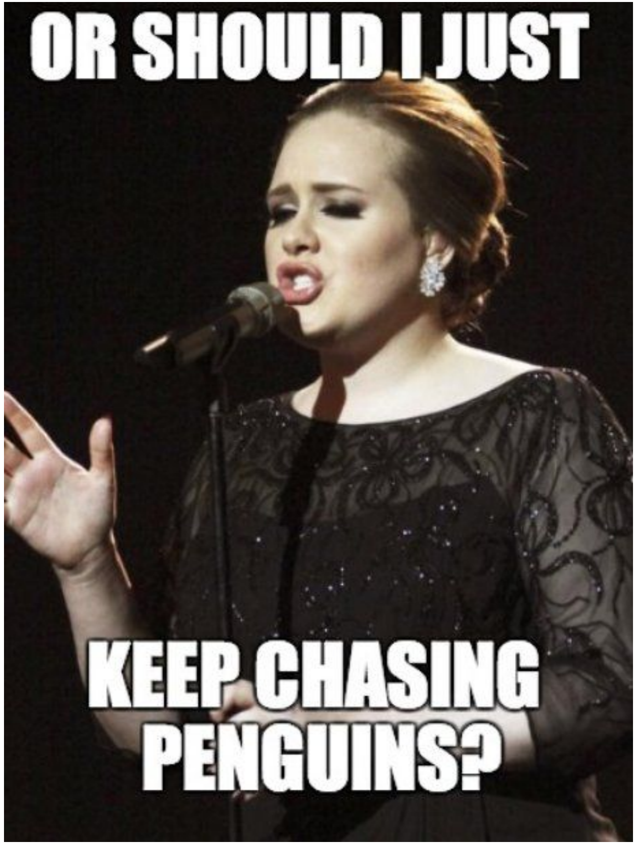


http://slazebni.cs.illinois.edu/spring17/lec26_audio.pdf

The “Classic Example”






OR SHOULD I JUST
KEEP CHASING PENGUINS?

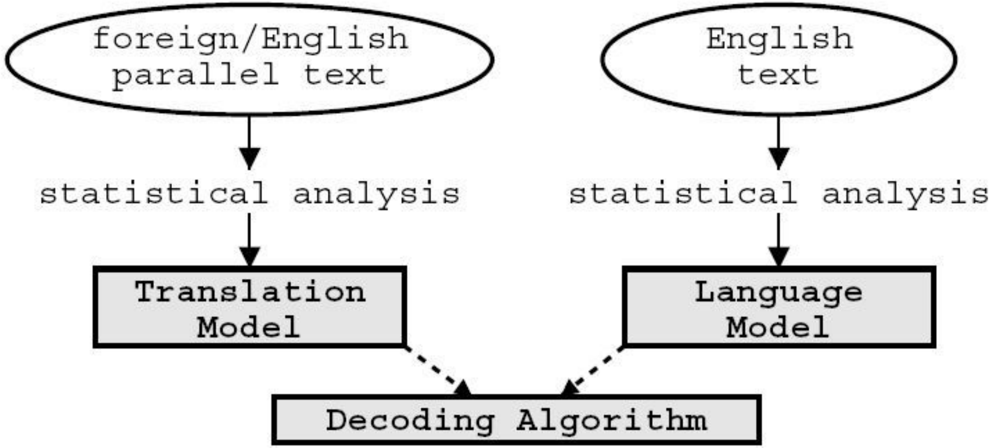

"I believe that the hot dogs go ooooooooonnn"


©NBC
Billie Jean McClung
@Billie_Jean26
Following
I like big butts and a can of limes. #misheardlyrics
Reply
Retweet
Favorite
More

The “Classic Example”

Statistical Machine Translation

- Components: Translation model, language model, decoder



Slide from Koehn 2008

The Simplest Language Models

"n-gram models"

1: unigram

2: bigram

3: trigram

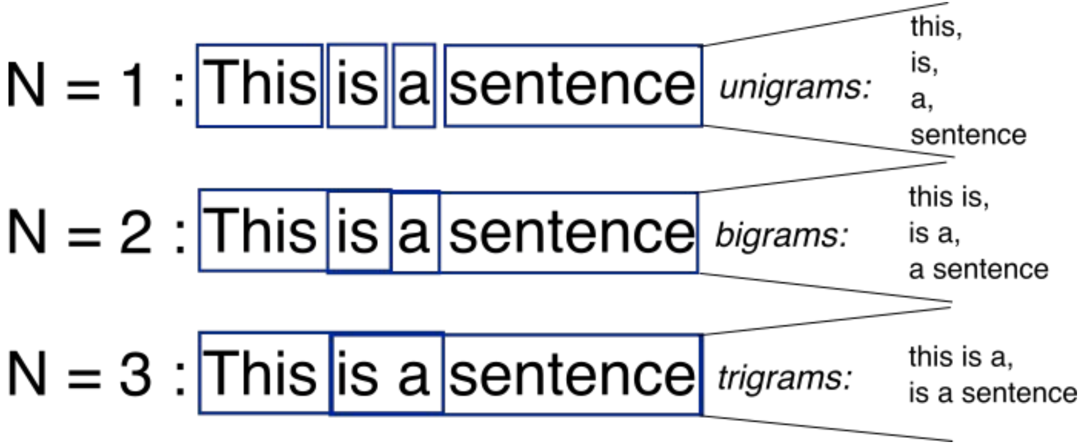
4: four-gram

5: five-gram

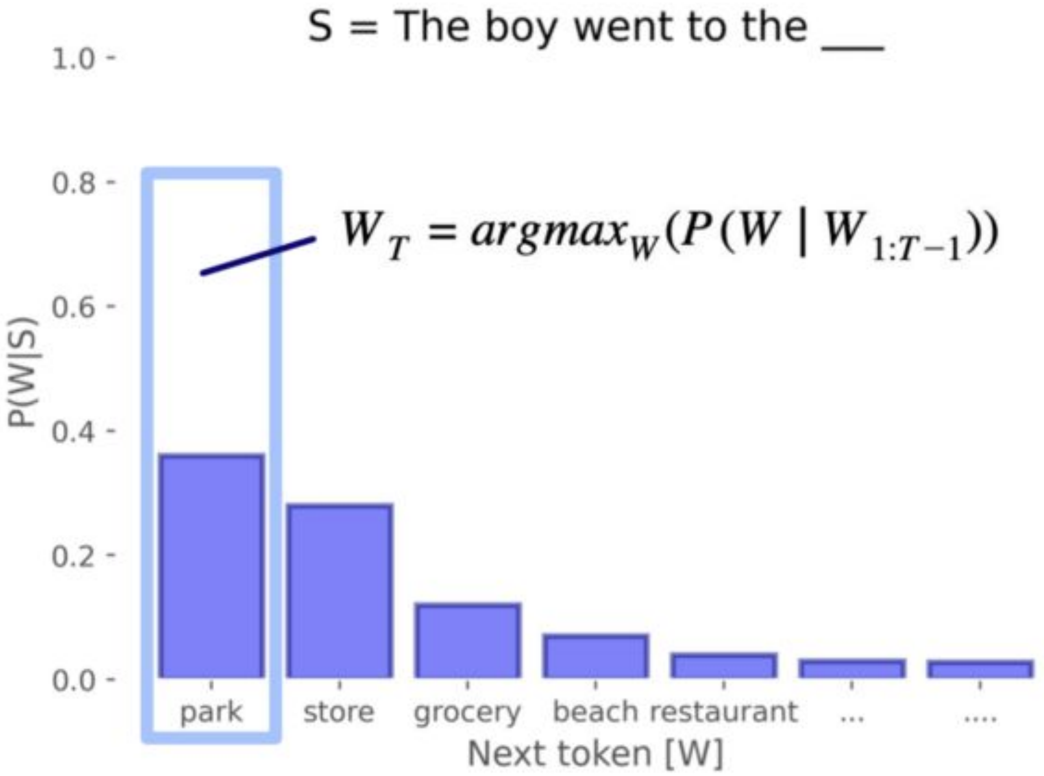
etc…

$$
\begin{aligned}
\mathrm{P}(X_4, X_3, X_2, X_1) &= \mathrm{P}(X_4 \mid X_3, X_2, X_1) \cdot \mathrm{P}(X_3, X_2, X_1) \\
&= \mathrm{P}(X_4 \mid X_3, X_2, X_1) \cdot \mathrm{P}(X_3 \mid X_2, X_1) \cdot \mathrm{P}(X_2, X_1) \\
&= \mathrm{P}(X_4 \mid X_3, X_2, X_1) \cdot \mathrm{P}(X_3 \mid X_2, X_1) \cdot \mathrm{P}(X_2 \mid X_1) \cdot \mathrm{P}(X_1)
\end{aligned}
$$

The Chain Rule of probability



The Simplest Language Models



https://blog.allenai.org/a-guide-to-language-model-sampling-in-allennlp-3b1239274bc3

Shall I compare thee to a *training corpus*?

1 gram	–To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have –Hill he late speaks; or! a more to leg less first you enter
2 gram	–Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. –What means, sir. I confess she? then all sorts, he is trim, captain.
3 gram	–Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, ’tis done. –This shall forbid it should be branded, if renown made it empty.
4 gram	–King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv’d in; –It cannot be but so.

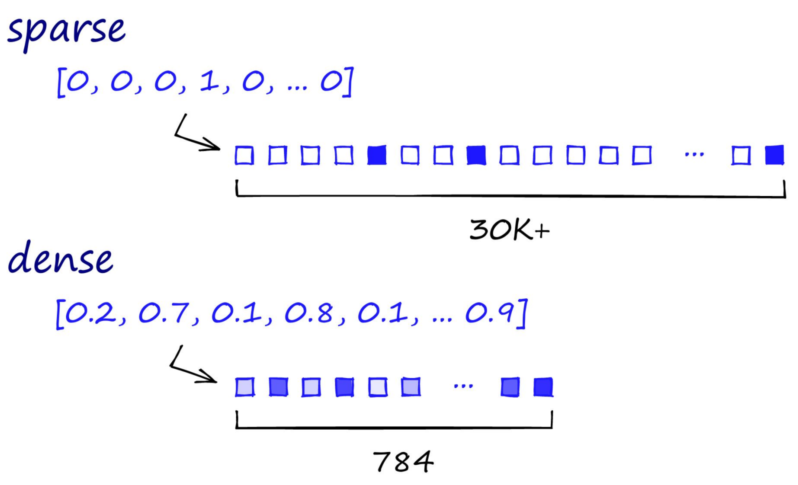
Problems with *n-gram* models

What are some issues you can think of relating to n-gram models?

- Unknown words
- Spelling mistakes
- Long distance dependencies
- Synonyms
- Data sparsity

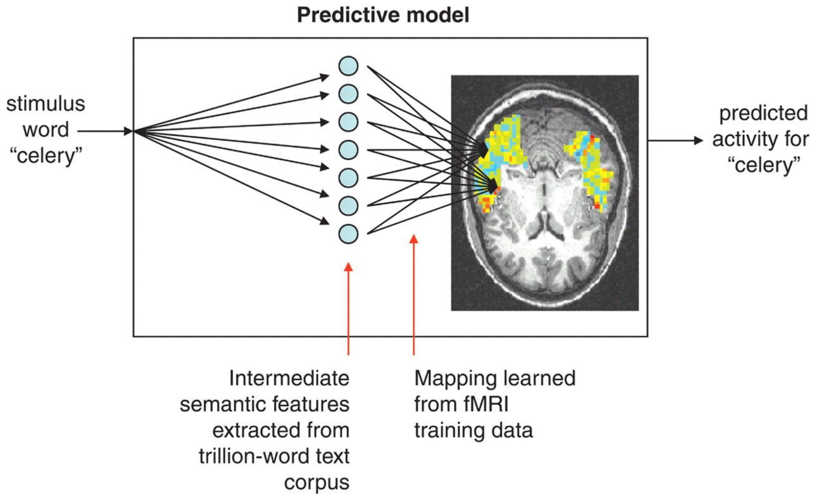
Embeddings ("dense representations")

- We need to be able to capture word meaning in a better way
- Three main methods were developed to achieve this
  - Co-occurrence statistics
  - Matrix factorisation methods
  - Neural network methods
- These (latter two methods) serve as key components in modern (L)LMs



Word Vectors via Co-occurrence Matrices

- A word's meaning is a function of the words it appears with
- This is known as the "**Distributional Hypothesis**"
- Word co-occurrences with common words not so helpful
- Co-occurrences with a specific subset of words is better



"You will know a word by the company it keeps"

J.R. Firth (1957)

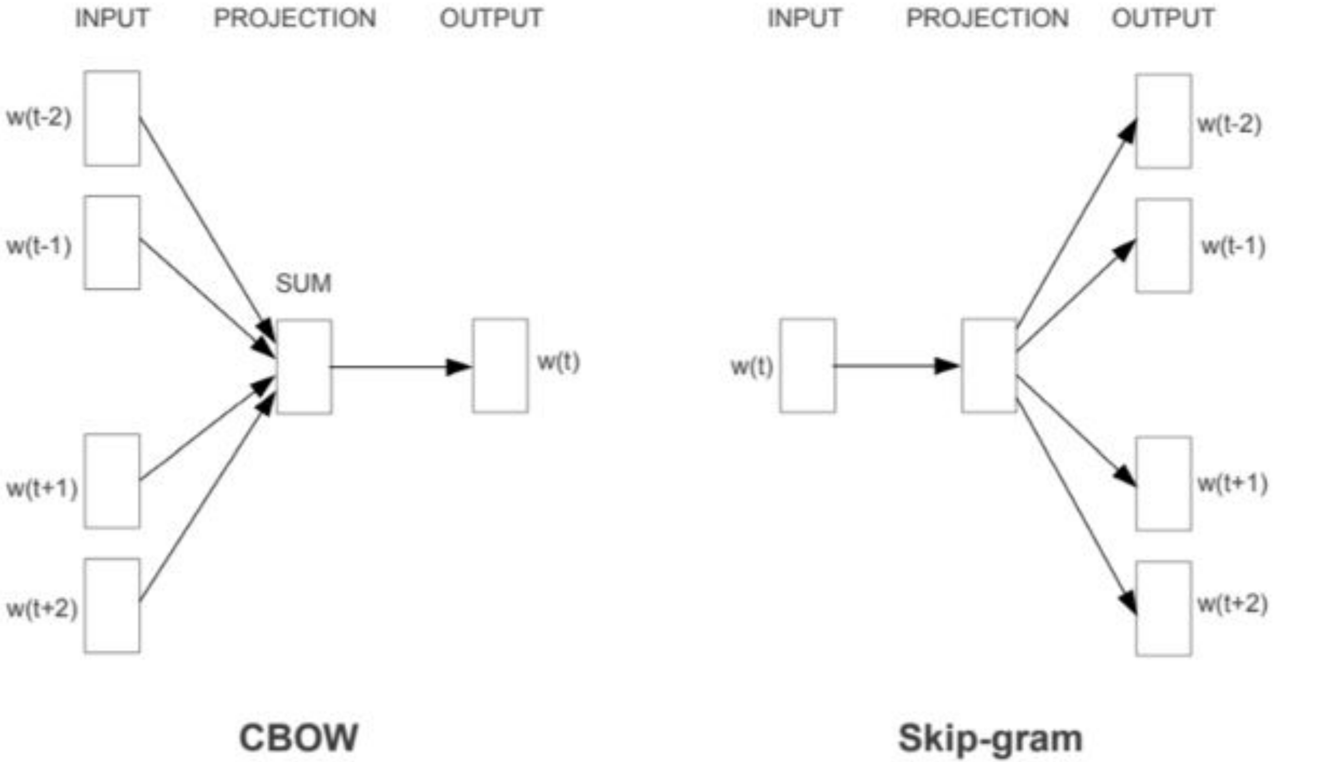
Word Vectors via Neural Networks

Word2Vec

- Scan a fixed-size window over a corpus of text
- For each window, pick the either the central word or all the context words
    - The choice of each leads to a slight variation of the Word2Vec algorithm
    - If you predict central word from context -> **CBOW**
    - If you predict context from central word -> **Skipgram**

http://jalammar.github.io/illustrated-word2vec/

Word2Vec



INPUT
PROJECTION
OUTPUT
w(t-2)
w(t-1)
SUM
w(t+1)
w(t+2)
w(t)
CBOW
INPUT
PROJECTION
OUTPUT
w(t)
w(t-2)
w(t-1)
w(t+1)
w(t+2)
Skip-gram

Word2Vec

Jay was hit by a ______ bus in...

by	a	red	bus	in

input 1	input 2	input 3	input 4	output
by	a	bus	in	red

Word2Vec (Skipgram)

Jay was hit by a red bus in...

by	a	red	bus	in

input	output
red	by
red	a
red	bus
red	in

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	in	the	…

input word	target word

Word2Vec (Skipgram)

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	in	the	…

input word	target word

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	in	the	…

thou	shalt	not	make	a	machine	in	the	…

input word	target word
not	thou
not	shalt
not	make
not	a

Word2Vec (Skipgram)

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	in	the	…

thou	shalt	not	make	a	machine	in	the	…

input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine

Word2Vec (Skipgram)

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	in	the	…

thou	shalt	not	make	a	machine	in	the	…

input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine

Word2Vec (Skipgram)

Thou shalt not make a machine in the likeness of a human mind

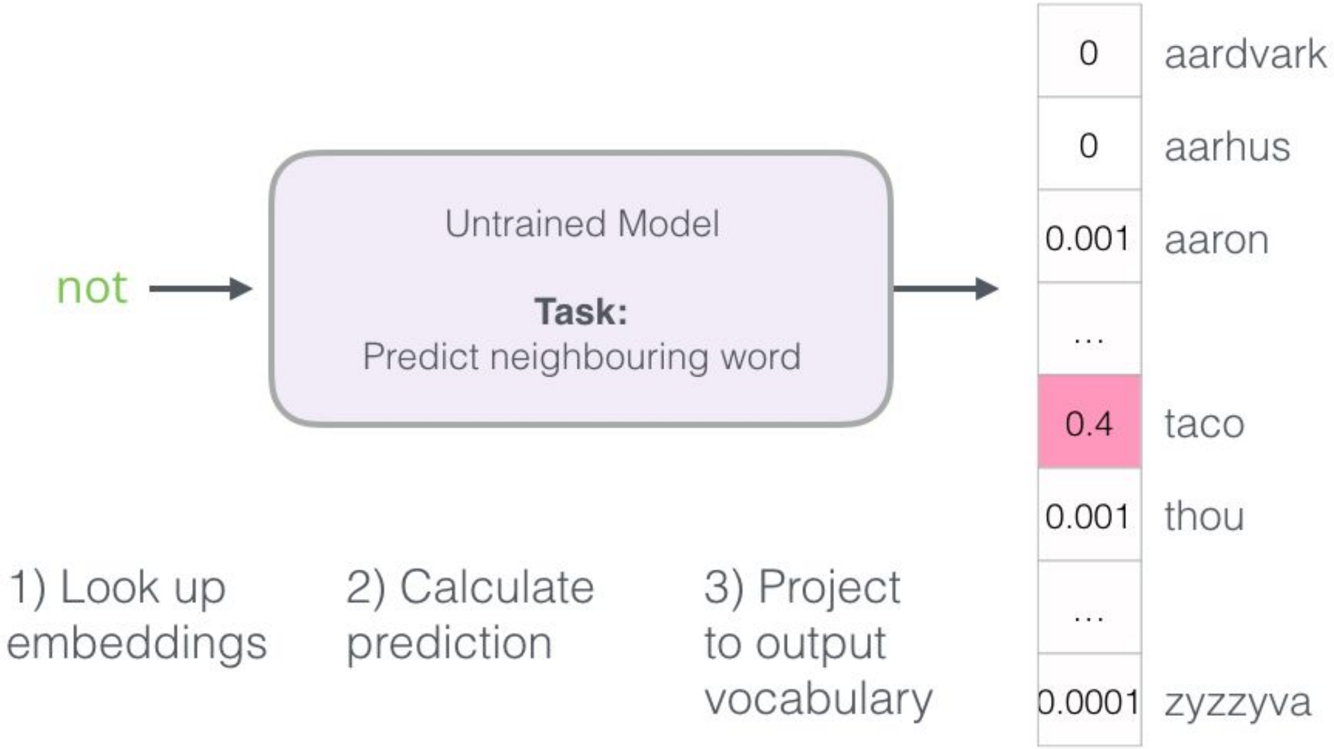
thou	shalt	not	make	a	machine	in	the	…
thou	shalt	not	make	a	machine	in	the	…
thou	shalt	not	make	a	machine	in	the	…
thou	shalt	not	make	a	machine	in	the	…
thou	shalt	not	make	a	machine	in	the	…

input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine
a	not
a	make
a	machine
a	in
machine	make
machine	a
machine	in
machine	the
in	a
in	machine
in	the
in	likeness

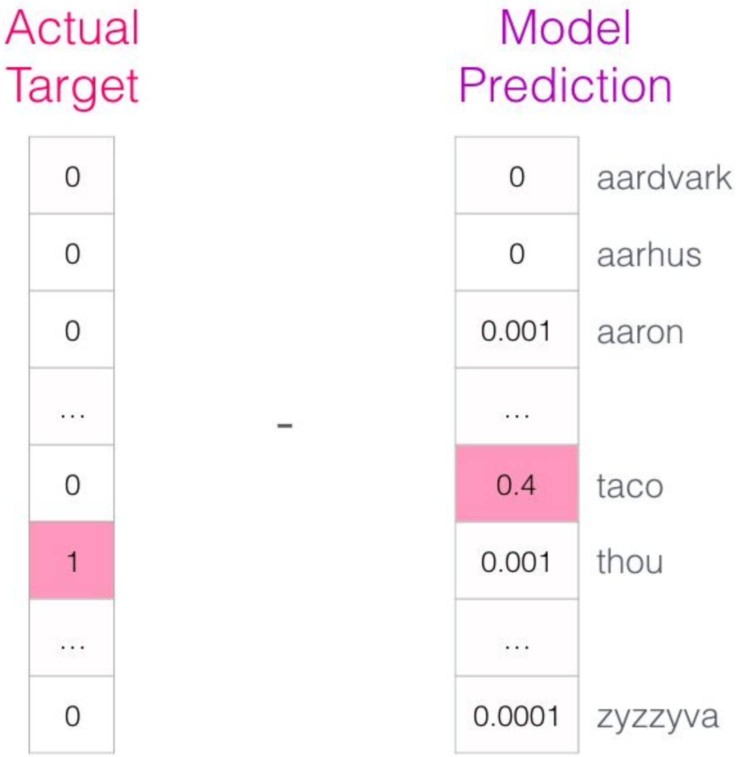
Word2Vec (Skipgram)

input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine
a	not
a	make
a	machine
a	in
machine	make
machine	a
machine	in
machine	the
in	a
in	machine
in	the
in	likeness

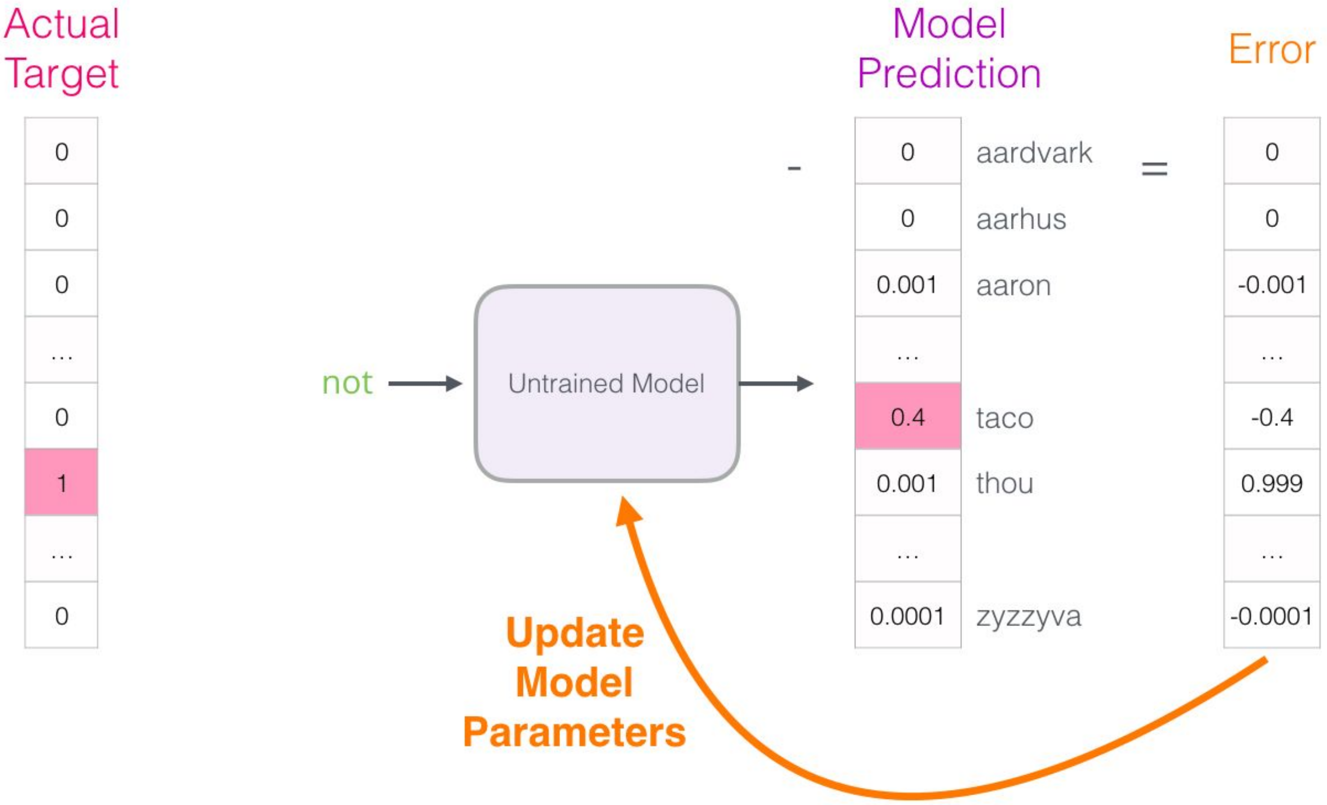
Word2Vec (Skipgram)



Word2Vec (Skipgram)



Word2Vec (Skipgram)



Word2Vec (Skipgram)

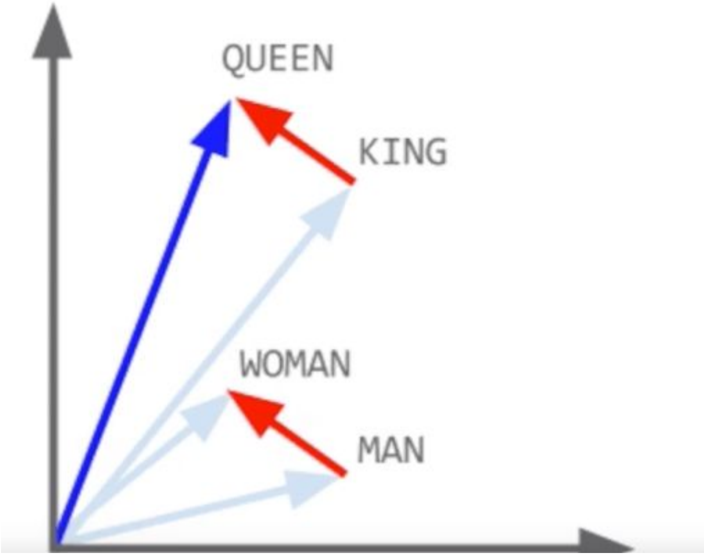
What size should the context window be?

Small window sizes → representations that are more interchangeable
Larger window sizes →representations that are semantically related

The idea is that **larger contexts** contain **more of a semantic domain** to influence a target word’s meaning

Word2Vec

- These word representations turned out to be interpretable
- They obey a certain representational geometry where subtraction of vectors encoded a semantic meaning that could be added to other words
- The closest vectors in the model to these dimensions were very often coherent



Word2Vec - *An aside*

- The representational geometry was not something we could have expected in advance
- This is absolutely not an "**obvious**" thing to happen (though in hindsight we pretend it was)
- The creator of word2vec, **Tomáš Mikolov,** recounts a funny story about this

> "*I wanted to convince my mentor at Microsoft that there was something interesting going on with the vector algebra, that* `king - man + woman = queen`*. I mentioned the idea and he didn't seem convinced at all that something like this could work. So, I asked him if he thought you could do plus / minus on the vectors and see correct results. He said "Of course not. That's completely stupid," and he was looking at me as if I had gone crazy. So, I took him to the computer and told him to look for the nearest neighbour and check the result. He was very surprised, but then got very excited and started thinking about how we could evaluate this in a more rigorous way.* " (My paraphrasing)

https://youtu.be/p-_0lOMfMYI?t=1275 (the interview)

Word Vectors via Matrix Factorisation

GloVe (Global Vector for Word Representation)

- Word-word co-occurrence matrix derived from a corpus
- Uses statistics from entire corpus (*not just local context windows*, hence: **global**)
- Factorisation of this matrix results in numerical word vectors

Intuition:

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	high	low	high	low
$P(k\|steam)$	low	high	high	low
$P(k\|ice)/P(k\|steam)$	> 1	< 1	~ 1	~ 1

https://nlp.stanford.edu/projects/glove/

Problems with (non-contextual) word embeddings

- The term “**context**” is used a lot for the previous algorithms
- However, those word representations are highly ***non-contextual***
- Words with one written form and multiple meanings (= **homonyms)** are processed the same
  - i.e. the same vector for ‘row’ will be updated in the following cases:
    - A huge **row** erupted after the results were revealed (argument)
    - Everybody line up in a **row** (a line / ordering)
    - The team were able to **row** for hours at a time (propel a boat with an oar)
- Words with opposite meanings (= **antonyms**) often appear in the same linguistic contexts
  - this means the representations are similar even though the meanings are opposite
  - is this always a bad thing? A relationship does exist between them ….

Model Evaluation Metrics

- How can we **evaluate** how good / bad a language model is?
- Depending on the goal:
  - **Extrinsic Evaluation**
  - **Intrinsic Evaluation**

Extrinsic Evaluation

- LM is a component in an NLP system
- You have different LM versions
- Switch them in the system and measure the change in performance
- Often computationally expensive

Intrinsic Evaluation

- More of a quick & easy method
- Requires independent test set
- Compare LMs on how well they predict the independent test set
- Common metric is **perplexity**

Model Evaluation Metrics

Perplexity

- Perplexity is the inverse probability of test set, normalised by the number of words in the test set

$$\mathrm{PP}(W) = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

- The probability of the test set is rearranged according to the chain rule of probability

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1 \dots w_{i-1})}}$$

- In this example, consider a bigram language model

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}}$$

See Jurafsky & Martin, Section 3.2.1 for more details

Model Evaluation Metrics

Perplexity

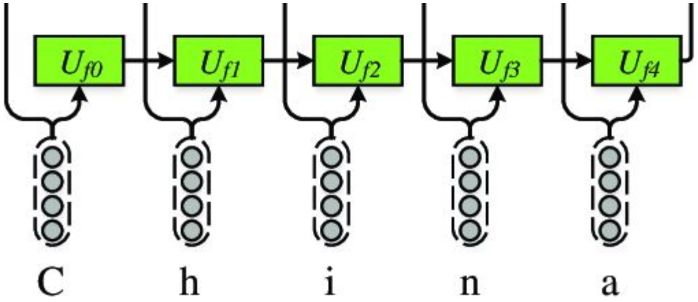
Finally, let's look at an example of how perplexity can be used to compare different n-gram models. We trained unigram, bigram, and trigram grammars on 38 million words (including start-of-sentence tokens) from the *Wall Street Journal*, using a 19,979 word vocabulary. We then computed the perplexity of each of these models on a test set of 1.5 million words with Eq. 3.16. The table below shows the perplexity of a 1.5 million word WSJ test set according to each of these grammars.

	Unigram	Bigram	Trigram
Perplexity	962	170	109

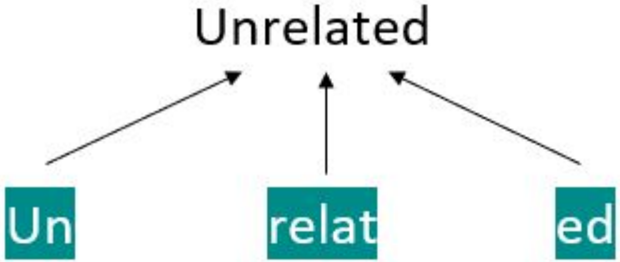
See Jurafsky & Martin, Section 3.2.1 for more details

Beyond “word” vectors

- Character-based models
  - Easily solve the “unknown word” problem
  - Lose a lot of information at higher levels, such as common morphemes in a language

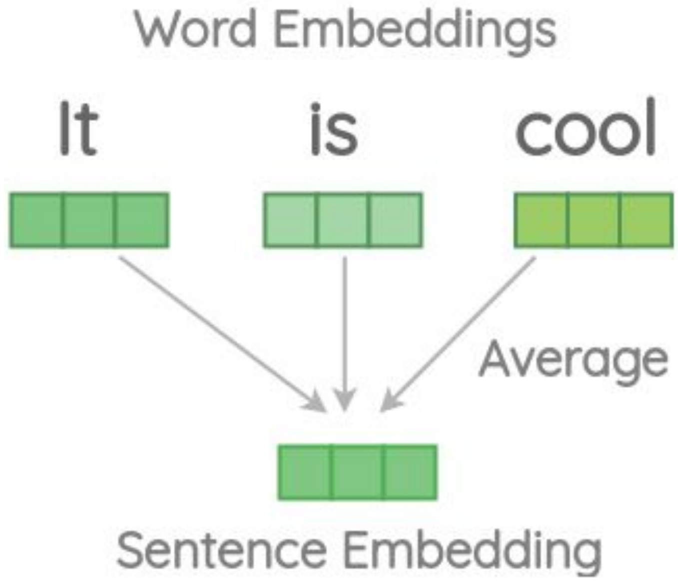


- Subword-based models
  - A compromise between word and character level models
  - Capture common morphemes and linguistically salient units, while also able to deal with novel vocabulary



Beyond “word” vectors

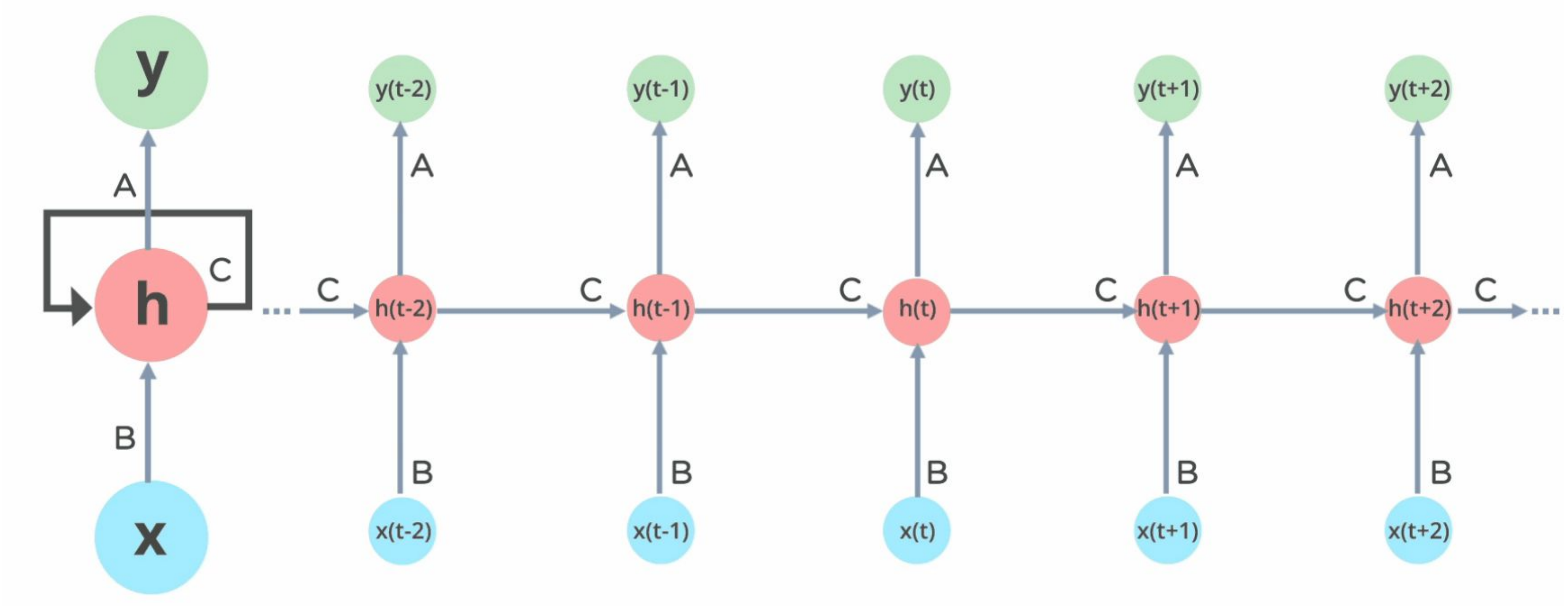
- How should sentences be represented?



Recurrent Neural Networks

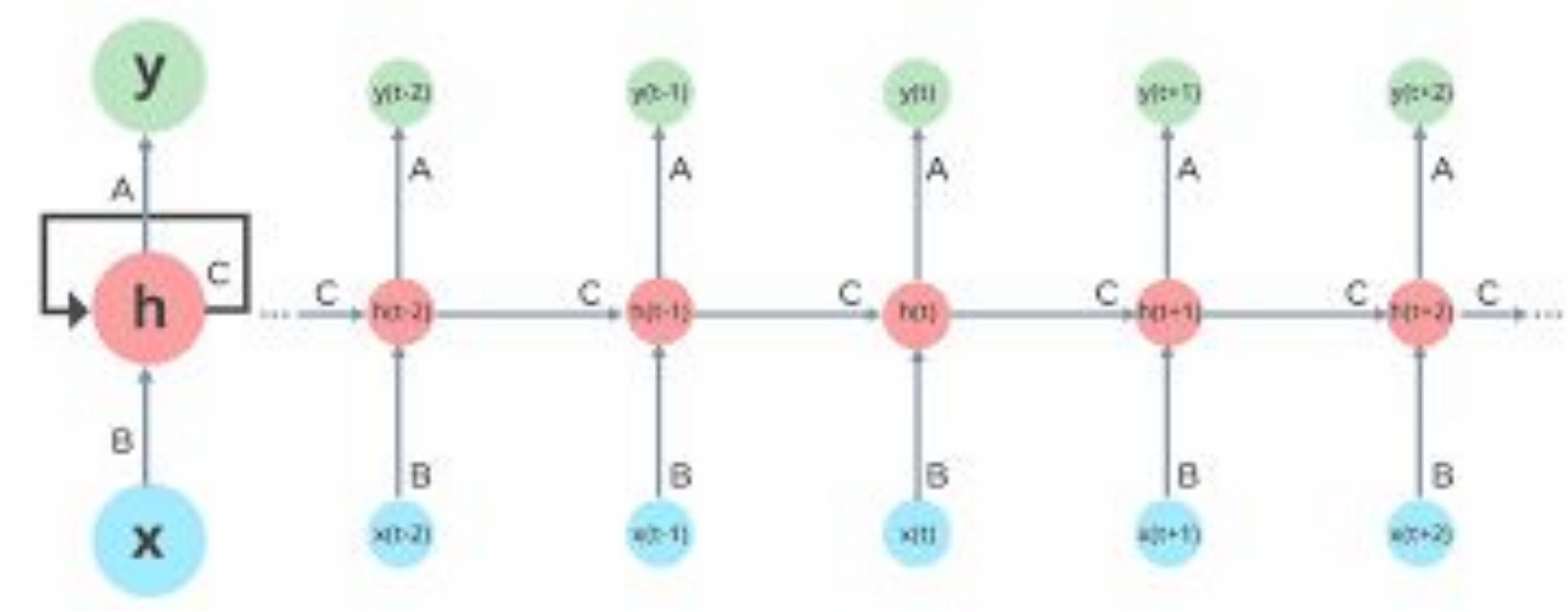
- Neural networks that could capture sequence information
- Language is (obviously) inherently sequential, so a perfect domain for RNNs
- RNNs also apply to many other time series analysis problems, too
- Idea is to extract a representation from a time step, then **give this output to the same network** as it takes in the input from the next time step in order to **incorporate the previous time step** representation (and so on and so forth until EOS)

Recurrent Neural Networks



https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn

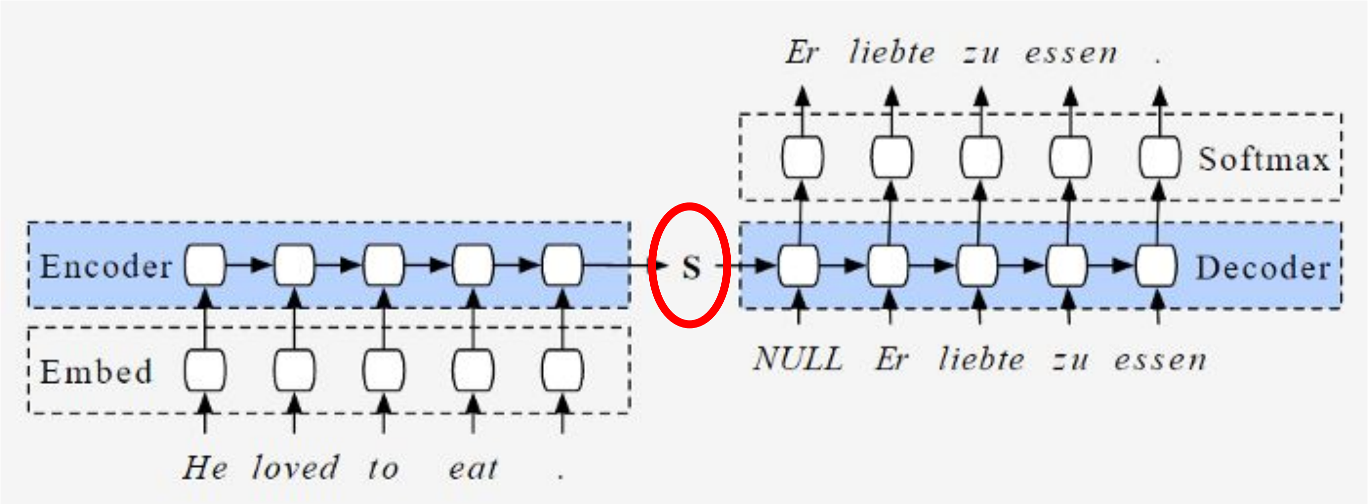
Recurrent Neural Networks



https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn

Sequence to Sequence Models (seq2seq)

- 2014 paper by Google ("Sequence to Sequence Learning with Neural Networks")
- Designed for Machine Translation
- Take 2 RNNs (one encoder and one decoder)
- Use one RNN to encode sentence and decode this vector with another RNN
- Bottleneck problem



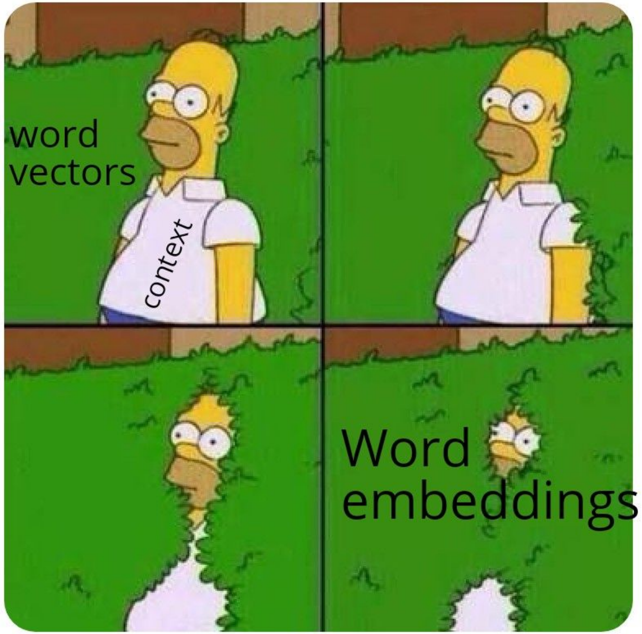
https://www.analyticsvidhya.com/blog/2020/08/a-simple-introduction-to-sequence-to-sequence-models/

Contextualised Word Embeddings

- Take original idea of word vectors
- Apply ideas from seq2seq models
- Reframe the language model problem
- Train models on giant corpora

(GPUs go brrrrrrr)

Word embeddings
in a nutshell



r/MLmemes

ELMo (2018)

“**Embeddings** from **Language** **Models**”

Started the whole **Sesame Street** craze

Use RNNs to create ***contextualised*** embeddings

Embeddings now seen as part of a **sequence**

Using seq2seq idea, develop a higher level of word embedding on top of traditional word vectors

How?

Train the seq2seq model as if it were a language model

ELMo: Why it’s one of the biggest advancements in NLP

Embeddings from Language Models (ELMo) is a state-of-the-art language modeling idea. What makes it so successful?

Published in 2018, “Deep Contextualized Word Embeddings” presented the idea of Embeddings from Language Models (ELMo), which achieved state-of-the-art performance on many popular tasks including question-answering, sentiment analysis, and named-entity extraction. ELMo has been shown to yield performance improvements of up to almost 5%. But what makes this idea so revolutionary?

ELMo (2018)

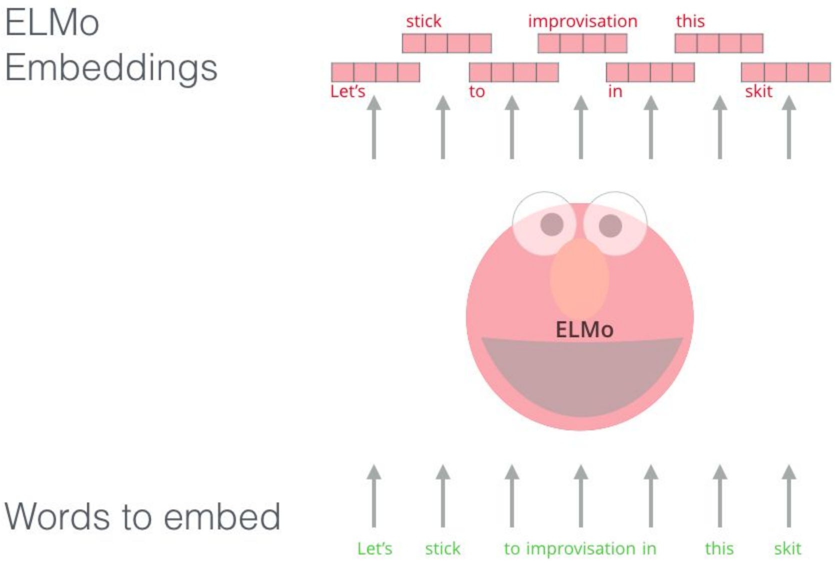

Hey ELMo, what's the embedding of the word "stick"?
There are multiple possible embeddings! Use it in a sentence.
Oh, okay. Here: "Let's stick to improvisation in this skit"
Oh in that case, the embedding is: -0.02, -0.16, 0.12, -0.1 ….etc

ELMo (2018)

- Character-based model
- Task-specific embeddings
- Trained to predict upcoming words on giant unlabeled text corpus
- Bidirectional model

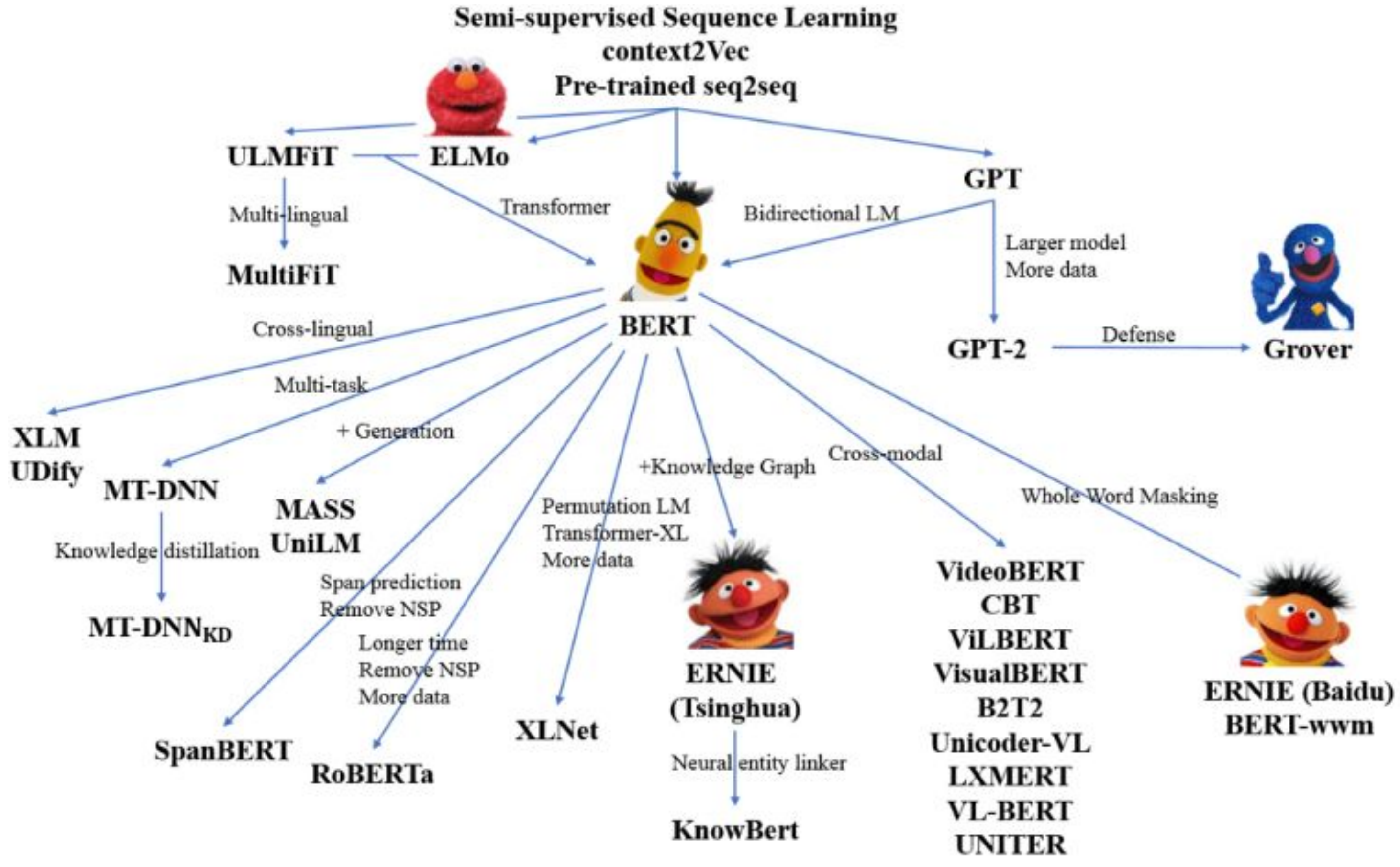
Basic Idea:

Instead of solving tasks directly with word vectors, use seq2seq framework to build contextualised word embeddings.

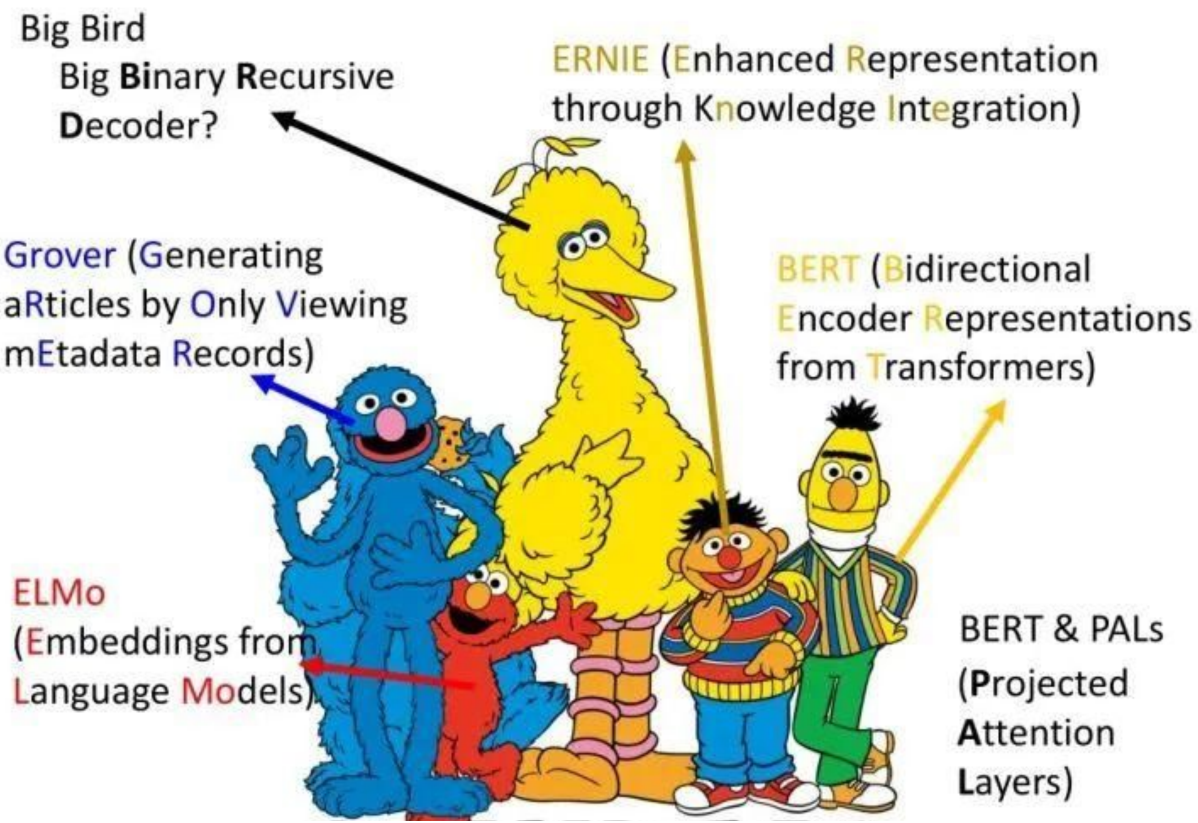


https://jalammar.github.io/illustrated-bert/

The Sesame Street Obsession



The Sesame Street Obsession



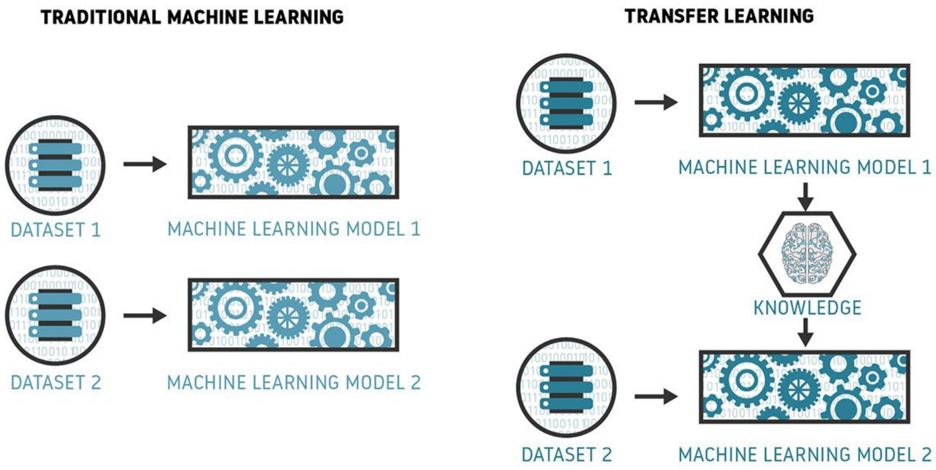
[1] Object-Semantics Aligned Pre-training

[2] The code and pre-trained models are re

Oscar

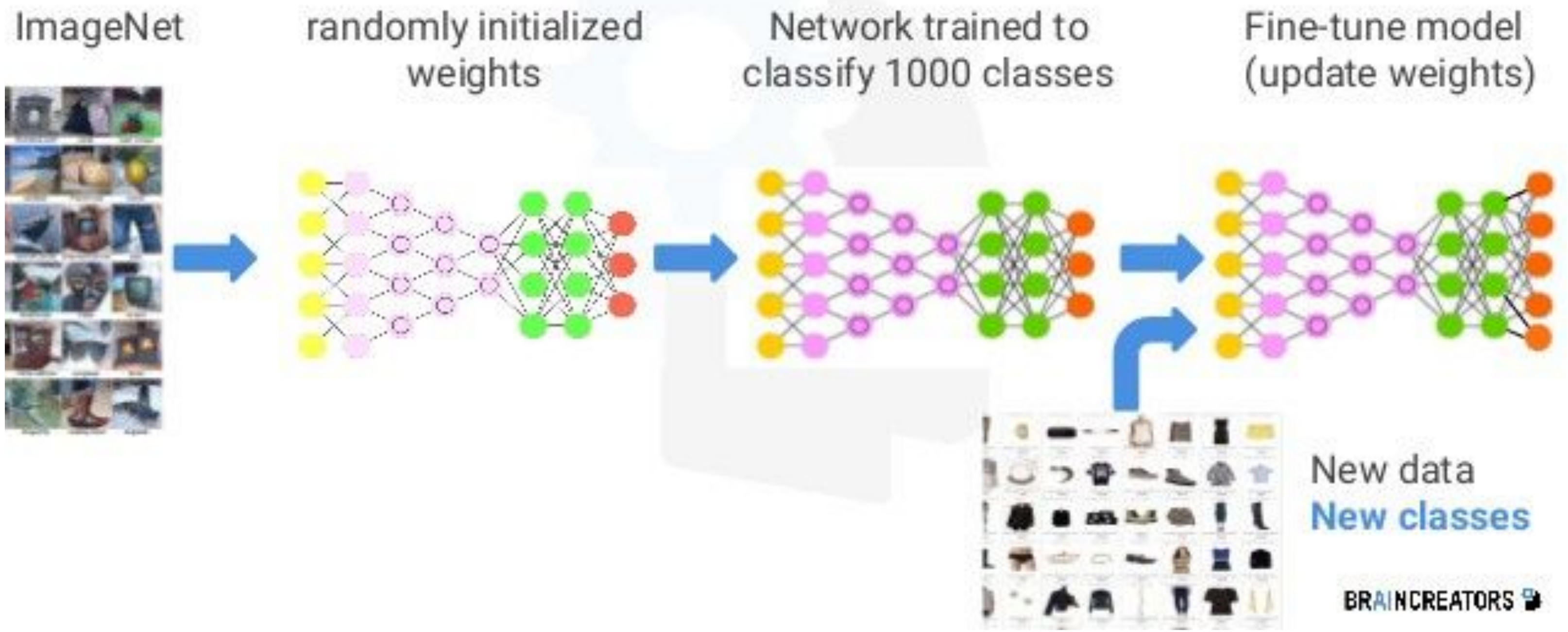
ULMFit (2018)

- Universal Language Model Fine-Tuning
- Introduced the idea of **transfer learning** to NLP
- This involved **pretraining** on general data
- Specified general techniques for **fine-tuning** in a generic sense
- Previous models all had a task-specific element to them

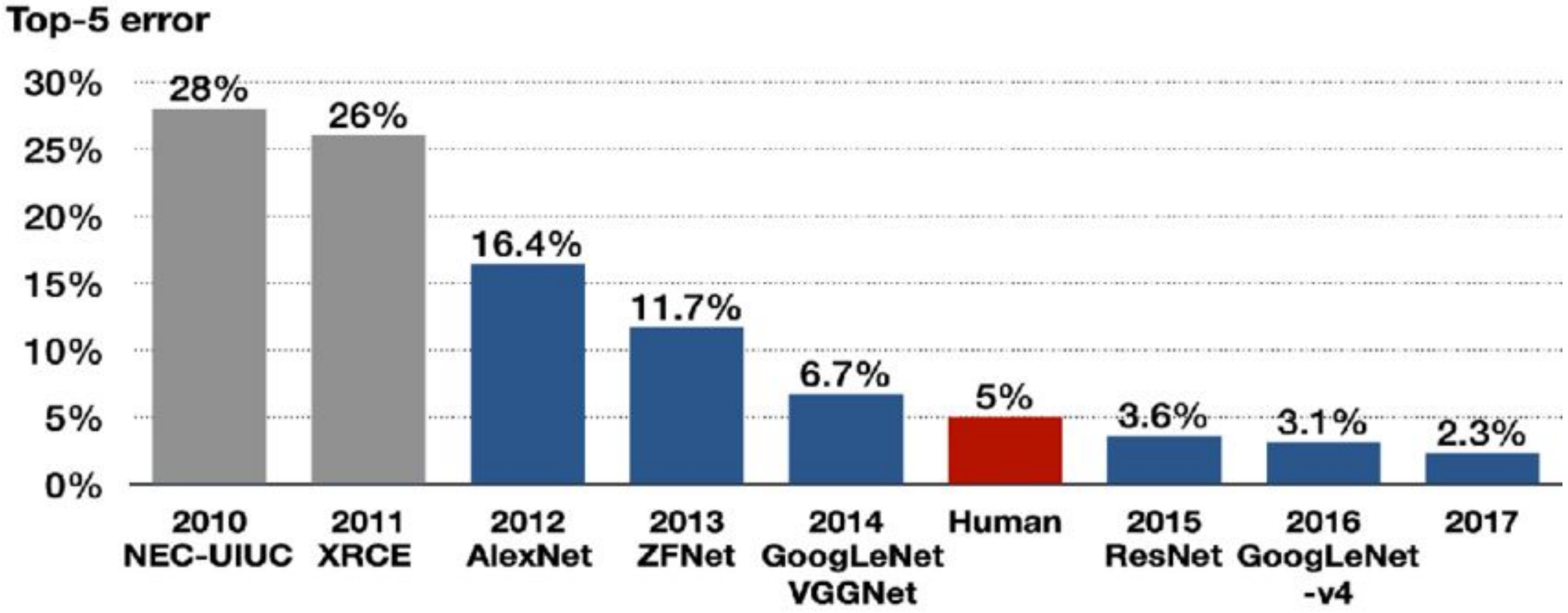


Universal Language Model Fine-tuning for Text Classification (Howard & Ruder, ACL 2018)

Transfer Learning in CV



NLP's "ImageNet" moment



NATURAL LANGUAGE PROCESSING

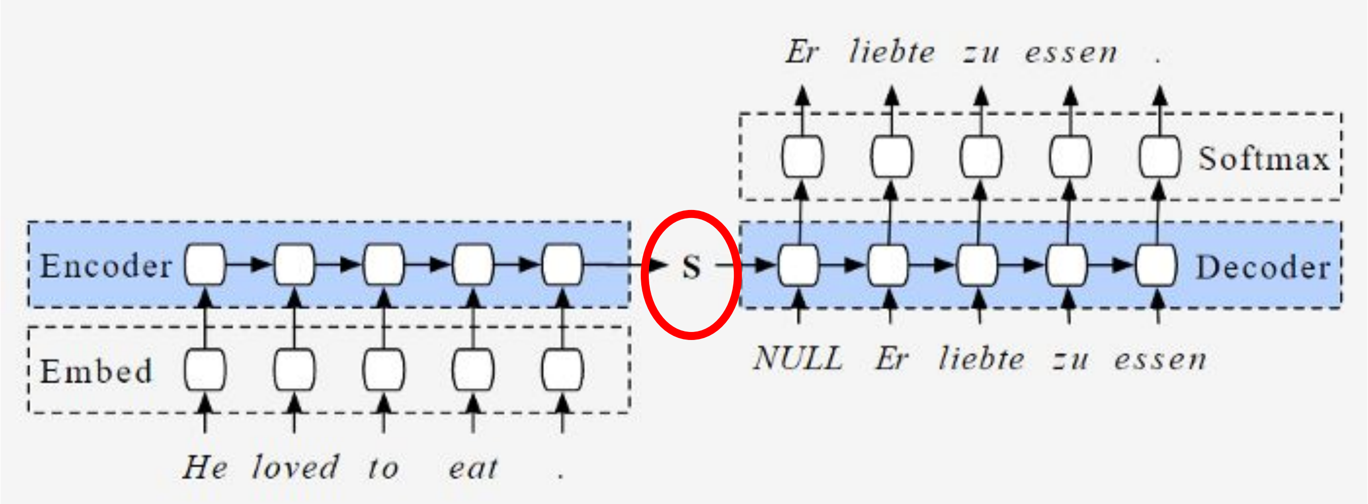
NLP's ImageNet moment has arrived

Big changes are underway in the world of NLP. The long reign of word vectors as NLP's core representation technique has seen an exciting new line of challengers emerge. These approaches demonstrated that pretrained language models can achieve state-of-the-art results and herald a watershed moment.

SEBASTIAN RUDER

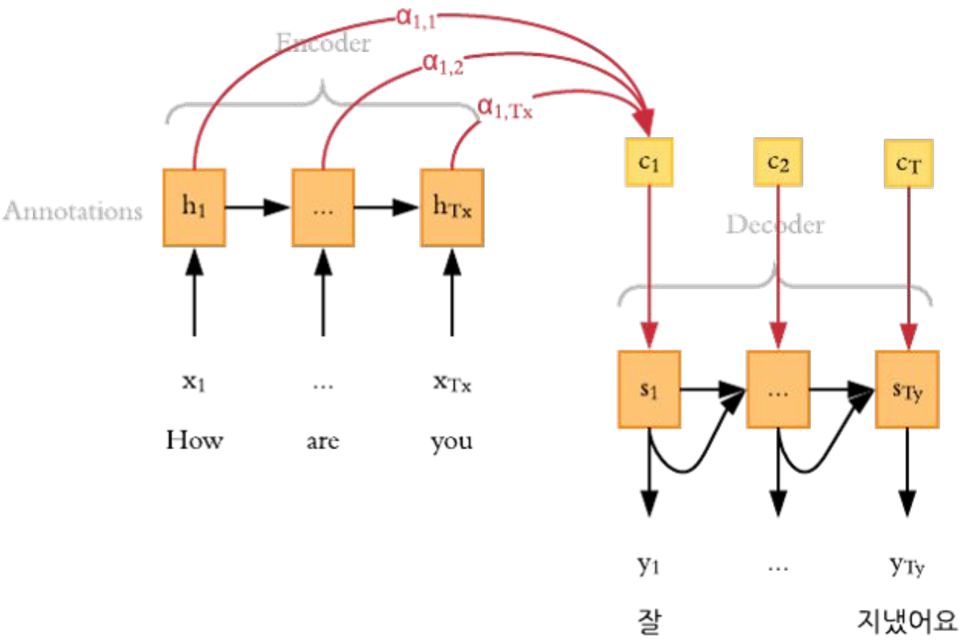
12 JUL 2018 • 16 MIN READ

RNNs Revisited



RNNs Revisited

- Attention mechanism
- Learn weights to multiply the encoder embeddings with so that the salient inputs are upweighted when passed to the decoder
- RNNs augmented with attention existed for a short time



Attention is all you Need (2017)

- << Rewind one year >>
- Removed the need for “**Recurrence**” in RNNs
- This dramatically improved training efficiency
- Positional embeddings represent input order
- Introduced “**Transformer**” architecture